

# Group Isomorphism with Fixed Subnormal Chains

Eugene M. Luks

Computer and Information Science Department  
University of Oregon

email: `luks@cs.uoregon.edu`

October 31, 2015

## Abstract

In recent work, Rosenbaum and Wagner showed that isomorphism of explicitly listed  $p$ -groups of order  $n$  could be tested in  $n^{\frac{1}{2} \log_p n + O(p)}$  time, roughly a square root of the classical bound. The  $O(p)$  term is entirely due to an  $n^{O(p)}$  cost of testing for isomorphisms that match fixed composition series in the two groups. We focus here on the fixed-composition-series subproblem and exhibit a polynomial-time algorithm that is valid for general groups. A subsequent paper will construct canonical forms within the same time bound.

## 1 Introduction

The complexity of testing isomorphism of groups of order  $n$  (input via Cayley tables) has long been cited as  $n^{\log_p n + O(1)}$ , where  $p$  is the smallest prime divisor of  $n$ ; this follows immediately from the fact that the group requires no more than  $\log_p n$  generators [11]. Wagner [22] suggested that this might be improved by a careful consideration of the isomorphisms that match fixed composition series. While composition series have long been a staple for such computational problems (e.g., [7]), Wagner's insight was that this could lead to an analyzable advantage in terms of a provable worst-case bound. Indeed, using that approach, Rosenbaum and Wagner [20] were able to improve the bound for  $p$ -groups to  $n^{(1/2) \log_p n + O(p)}$ . The  $O(p)$  term is contributed by their  $n^{O(p)}$  complexity analysis of fixed-composition-series isomorphism in  $p$ -groups. Their main result then uses the fact that there are  $n^{(1/2) \log_p n + O(1)}$  composition series to consider. Rosenbaum [18] extended the result to solvable groups, achieving an  $n^{(1/2) \log_p n + O(\log n / \log \log n)}$  time for isomorphism, the fixed-composition series subproblem contributing to the  $\log n / \log \log n$  term. Making use of a

canonical-form version of the fixed-composition-series subproblem, Rosenbaum [19] subsequently showed that a “collision” method yields a square-root improvement. That result is striking and there is much to be appreciated in the methods. However, the composition-series-isomorphism subproblem remains appealing in its own right and it appears to be susceptible to established algebraic and computational methods.

Thus, we focus on the following problem.

#### COMPSERIESISO

GIVEN: Groups  $G_1, G_2$  given by Cayley tables;

composition series

$$G_1 = G_{1,0} \triangleright G_{1,1} \triangleright G_{1,2} \triangleright \cdots \triangleright G_{1,m} = \mathbf{1},$$

$$G_2 = G_{2,0} \triangleright G_{2,1} \triangleright G_{2,2} \triangleright \cdots \triangleright G_{2,m} = \mathbf{1}.$$

QUESTION: Is there an isomorphism  $f : G_1 \rightarrow G_2$  such that

$$f(G_{1,i}) = G_{2,i}, \text{ for } 0 \leq i \leq m?$$

The treatments of COMPSERIESISO for nilpotent and solvable groups in [22], [20], [18], put great effort into reductions to instances of bounded-valence graph isomorphism so as to plug in the main result of [12]. However, underlying the latter was a method for set-stabilizers in permutation groups which we can apply directly in a natural approach to COMPSERIESISO. This results in both a better bound and an extension to general groups. Specifically,

**Theorem 1.1** *COMPSERIESISO is in polynomial time.*

If one separates the elements needed for just the solvable-group case, the proof can be compressed to a few lines.

It should be no surprise that the method actually returns *all* isomorphisms in the form of a coset of the analogous automorphism group. In fact, our discussion concentrates mostly on finding automorphism groups.

#### COMPSERIESAUTO

GIVEN: A group  $G$  given by Cayley table;

a composition series  $G = G_0 \triangleright G_1 \triangleright G_2 \triangleright \cdots \triangleright G_m = \mathbf{1}$ .

FIND: (Generators for) the group  $\{f \in \text{Aut}(G) \mid f(G_i) = G_i \text{ for } 0 \leq i \leq m\}$ .

We prove

**Theorem 1.2** *COMPSERIESAUTO is in polynomial time.*

The application to isomorphism then follows a quick and standard path.

An immediate consequence of Theorem 1.1 is the time bound  $n^{(1/2)\log_p n + O(1)}$  for testing isomorphism of groups of order  $n$ , where  $p$  is the smallest prime divisor of  $n$ ; this is due to the upper bound of  $n^{(1+\log_p n)/2}$  on the number of composition series (credited to Babai in [20, Lemma 3.1]). Rosenbaum [19] has already realized that timing for isomorphism using “bidirectional collision”, though his method comes at a substantial cost in space. Collision and other innovative methods in [19] also mesh well with our results, with further implications for group isomorphism. We will be pleased to borrow from those methods in a follow-up paper which will first extend our COMPSERIESISO algorithm to the computation of canonical forms.

We remark that Babai [2] has found a holomorph approach to COMPSERIESISO which also improves the earlier bounds and is in polynomial time for solvable groups.

Our method for COMPSERIESAUTO involves repeated consideration of a classic issue in automorphism-group computation. Specifically, for  $H \trianglelefteq G$ , we are given  $\mathcal{A} \leq \text{Aut}(G/H)$  and  $\mathcal{B} \leq \text{Aut}(H)$ , and we want to determine the pairs  $(\alpha, \beta) \in \text{Aut}(G/H) \times \text{Aut}(H)$  that “lift” to automorphisms of  $G$ , if any such exist. The obstruction to such lifting is easy to formulate algebraically and, in general, it poses a difficult computational problem. However, for the  $\mathcal{A}$  and  $\mathcal{B}$  that arise herein, we are able to view the obstruction in stages, each of which is resolvable using methods that are in polynomial time for solvable permutation groups. We give two procedures for this. An elementary method, described in §2.2, is all one needs for the resolution of COMPSERIESAUTO for solvable groups, but it is not effective for general groups. The method is then strengthened in §2.3 to one that is generally applicable.

In §3, we recall the divide-and-conquer method for set-stabilizer in permutation groups and show that it is in polynomial time for the groups we encounter since they are shown to have solvable subgroups of “small” index.

There are two demonstrations of Theorem 1.2 in §4. They exhibit two ways of breaking the problem down into instances of the “lift” scenario that are amenable to a set-stabilizer approach (assuming they are not already in polynomial time via brute-force enumeration).

Theorem 1.1 is succinctly resolved in §5 by viewing it as an extended application of the methods of §4.

## Notation.

Suppose  $G$  is a group acting on the set  $\Omega$ . For  $g \in G$ , we denote the image of  $\alpha \in \Omega$  under the action of  $g$  by  $\alpha^g$ . For  $\Delta \subseteq \Omega$ , let  $G_\Delta = \{g \in G \mid \Delta^g = \Delta\}$ .

The automorphism group of  $G$  is denoted by  $\text{Aut}(G)$ , which we view as a subgroup of  $\text{Sym}(G)$ . Thus, for  $H \leq G$ ,  $\text{Aut}(G)_H$  denotes the group of automorphisms stabilizing (or normalizing)  $H$ .

Implicit in the statement that  $G$  is given by a Cayley table is the assumption that the elements of  $G$  can be listed in polynomial time. Permutation groups are assumed to be input or output via generating sets. A coset of a permutation group  $J$  is input or output via generators for  $J$  and a single representative.

For other concepts and notation, we refer the reader to [6]. For background on polynomial-time computability in permutation groups, see [10], [12], [15], [21].

## 2 The key subproblem

Throughout this section, we assume  $G$  is given by a Cayley table and  $H \trianglelefteq G$ . Then  $\text{Aut}(G)_H$  can be viewed as a subgroup of the wreath product  $\text{Sym}(H) \wr \text{Sym}(G/H)$ , and especially a subgroup of  $\text{Sym}(H) \wr \text{Sym}(G/H)_{\{H/H\}}$ . (The subscript  $\{H/H\}$  signifies that we fix this single “point” in the permutation domain  $G/H$ .) For the reader’s convenience we give a more explicit indication of the latter group, namely,

$$\text{Sym}(G)_{H,G/H} := \{\gamma \in \text{Sym}(G)_H \mid \gamma \text{ permutes the cosets of } H\}.$$

There is a natural homomorphism

$$\Theta_{G,H} : \text{Sym}(G)_{H,G/H} \rightarrow \text{Sym}(G/H) \times \text{Sym}(H).$$

Inasmuch as  $H \trianglelefteq G$  will always be clear in context, we let  $\Theta := \Theta_{G,H}$ . For  $x \in G$ , we let  $\iota(x)$  denote the restriction to  $H$  of the inner automorphism due on  $x$ , i.e., for  $h \in H$   $h^{\iota(x)} = x^{-1}hx$ ; this is extended to  $X \subseteq G$  by  $\iota(X) = \{\iota(x) \mid x \in X\}$ . Because of repeated usage, it is useful to set  $C := \{g \in G \mid \forall h \in H, gh = hg\}$ , i.e., the centralizer of  $H$  in  $G$ .

### 2.1 Lifting automorphisms from $G/H$ and $H$

Noting that  $\Theta(\text{Aut}(G)_H) \leq \text{Aut}(G/H) \times \text{Aut}(H)$ , we are concerned with the following problem.

**AUTLIFTING**

**GIVEN:**  $H \trianglelefteq G$ ;  $\mathcal{A} \leq \text{Aut}(G/H)$ ;  $\mathcal{B} \leq \text{Aut}(H)$ .

**FIND:**  $\text{Aut}(G)_H \cap \Theta^{-1}(\mathcal{A} \times \mathcal{B})$ .

There are a couple of natural ways to reduce Theorem 1.2 to polynomial-time instances of AUTLIFTING. We will describe these in §4.1,4.2.

The reader may already recognize AUTLIFTING as a frequent issue in studies of group automorphism/isomorphism, theoretical or computational (see, e.g., [8, §8.9]). Special

cases are attacked with varied machinery and success. Our method is guided by properties of the relevant groups that enable polynomial-time steps.

Our approaches to AUTLIFTING each involve defining a group  $\mathcal{L}$  such that

- $\text{Aut}(G)_H \leq \mathcal{L} \leq \text{Sym}(G)_{H,G/H}$ .
- It is “easy” to find  $\Theta(\mathcal{L}) \cap (\mathcal{A} \times \mathcal{B})$ .
- It is “easy” then to lift to  $\widehat{\mathcal{L}} = \mathcal{L} \cap \Theta^{-1}(\mathcal{A} \times \mathcal{B})$ .
- It is “easy” to find  $\widehat{\mathcal{L}} \cap \text{Aut}(G)$ .

## 2.2 First choice of $\mathcal{L}$

We consider the following supergroup of  $\text{Aut}(G)_H$ .

$$\mathcal{L}_1 = \{\gamma \in \text{Sym}(G)_{H,G/H} \mid \forall g \in G, h \in H : (hg)^\gamma = h^\gamma g^\gamma\}.$$

### 2.2.1 Theory

#### Lemma 2.1

- (i) *The kernel of  $\Theta|_{\mathcal{L}_1}$  is isomorphic to the direct product of  $|G/H| - 1$  copies of  $H$ .*
- (ii) *(Generators for)  $\text{Ker}(\Theta|_{\mathcal{L}_1})$  can be found in polynomial time.*

PROOF: Let  $\gamma \in \text{Ker}(\Theta|_{\mathcal{L}_1})$ . Then  $\forall h \in H, h^\gamma = h$  and  $\forall x \in G, (Hx)^\gamma = Hx$ . Consider the action induced by  $\gamma$  on  $Hg \neq H$ . We have  $g^\gamma = gk$  for some  $k \in H$  (since  $gH = Hg$ ). Then for any  $hg \in Hg$ ,  $(hg)^\gamma = h^\gamma g^\gamma = h g k$ , that is,  $\forall x \in gH, x^\gamma = xk$ . Thus, the action of  $\text{Ker}(\Theta|_{\mathcal{L}_1})$  on each  $Hg \neq H$  is the group of right multiplications by  $H$ , a group isomorphic to  $H$ . Since the actions are independent across the cosets, (i) follows.

For (ii), we deal with each  $Hg \neq H$  independently. Focussing on a coset  $X$ , we form for each  $h \in H$ , the permutation  $\phi_{X,h}$  of  $G$  that fixes all  $x \notin X$  and maps  $x \mapsto xh$  for  $x \in X$ . Then  $\text{Ker}(\Theta|_{\mathcal{L}_1})$  is generated by  $\{\phi_{X,h} \mid h \in H, X \in G/H \text{ with } X \neq H\}$ . Take the union of all these permutations for all  $Hg \neq H$ . (A bit more economically, only use  $h$  in a generating set for  $H$ .)  $\square$

**Lemma 2.2**  $\text{Aut}(G/H) \times \text{Aut}(H) \leq \Theta(\mathcal{L}_1)$ . Furthermore, given  $(\alpha, \beta) \in \mathcal{A} \times \mathcal{B}$ ,  $\mathcal{L}_1 \cap \Theta^{-1}(\alpha, \beta)$  can be constructed in polynomial time.

PROOF: It suffices to show, for any  $\alpha \in \text{Aut}(G/H)$ ,  $\beta \in \text{Aut}(H)$ , we can construct a *single*  $\gamma \in \mathcal{L}_1$ , for which  $\Theta(\gamma) = (\alpha, \beta)$  since, by Lemma 2.1 the coset  $\gamma \text{Ker}(\Theta|_{\mathcal{L}_1})$  then comprises the set of preimages of  $(\alpha, \beta)$  in  $\mathcal{L}_1$ .

We construct  $\gamma \in \mathcal{L}_1$  as follows. For  $h \in H$ , define  $h^\gamma := h^\beta$ . For each coset of  $X$  of  $H$  in  $G$  with  $X \neq H$ , define  $\gamma$  on  $X$  as follows:

1. Fix any  $a \in X$  (so  $X = Ha$ ).
2. Fix any  $b \in X^\alpha$ .
3. For all  $h \in H$ , define  $(ha)^\gamma := h^\beta b$ .

□

Using Lemmas 2.1 and 2.2, we conclude

**Proposition 2.3**

- (i) Given  $\mathcal{A} \leq \text{Aut}(G/H)$  and  $\mathcal{B} \leq \text{Aut}(H)$ ,  $\widehat{\mathcal{L}}_1 := \mathcal{L}_1 \cap \Theta^{-1}(\mathcal{A} \times \mathcal{B})$  can be constructed in polynomial time.
- (ii)  $\widehat{\mathcal{L}}_1$  is an extension of  $\mathcal{A} \times \mathcal{B}$  by a direct product of copies of  $H$ .

PROOF: For (i),  $\widehat{\mathcal{L}}_1$  is generated by the lifts of the generators of  $\mathcal{A} \times \mathcal{B}$  together with generators of  $\text{Ker}(\Theta|_{\mathcal{L}_1})$ . □

**2.2.2 Algorithm**

**Step 1.**  $\widehat{\mathcal{L}}_1 := \mathcal{L}_1 \cap \Theta^{-1}(\mathcal{A} \times \mathcal{B})$

METHOD: By Lemma 2.3(i), this is in polynomial time for any  $\mathcal{A}, \mathcal{B}$ . □

**Step 2.** Find  $\{\gamma \in \widehat{\mathcal{L}}_1 \mid \gamma \in \text{Aut}(G)\}$ .

METHOD: This can be expressed as a set-stabilizer problem for the natural extension of  $\widehat{\mathcal{L}}_1 \leq \text{Sym}(G)$  to an action on  $G \times G \times G$ . The set to stabilize is  $\{(a, b, ab) \mid a, b \in G\}$ . □

**Remark 2.4** In an inductive approach to COMPSERIESAUTO for solvable groups the calls to AUTLIFTING result in a solvable  $\widehat{\mathcal{L}}_1$ , thus putting Step 2 in polynomial time. (We provide this forecast in the expectation that some readers would like to finish the solvable case as an exercise.) A more restrictive  $\mathcal{L}$  will yield our main results for general groups, thus making this section superfluous. Nevertheless, we retain this discussion of  $\mathcal{L}_1$ . By switching back to  $\mathcal{L}_1$  in §4.2 whenever  $H$  is solvable, we limit the requisite machinery to the early paper [12].

## 2.3 Second choice of $\mathcal{L}$

Consider now a more restricted supergroup of  $\text{Aut}(G)_H$ .

$$\mathcal{L}_2 = \{ \gamma \in \text{Sym}(G)_{H, G/H} \mid \forall g \in G, h \in H, (hg)^\gamma = h^\gamma g^\gamma \text{ and } (gh)^\gamma = g^\gamma h^\gamma \}.$$

### 2.3.1 Theory

The advantage of cutting down from  $\mathcal{L}_1$  to  $\mathcal{L}_2$  is that  $\text{Ker}(\Theta|_{\mathcal{L}_2})$  is abelian, which plays a role in guaranteeing polynomial-time set-stabilizers for the instances of  $\widehat{\mathcal{L}}_2$  that arise. (Actually, we only need that kernel to be solvable.) However, since  $\mathcal{A} \times \mathcal{B}$  may not be contained in  $\Theta(\mathcal{L}_2)$ , we will first have to determine the liftable subgroup. We are guided in this by some properties of  $\Theta(\mathcal{L}_2)$ .

**Lemma 2.5** *Let  $\gamma \in \mathcal{L}_2$ . Then for  $g \in G, h \in H, (g^{-1}hg)^\gamma = (g^\gamma)^{-1}h^\gamma g^\gamma$ .*

PROOF: For  $g \in G, h \in H, \gamma \in \mathcal{L}_2$ ,

$$g^\gamma (g^{-1}hg)^\gamma = (gg^{-1}hg)^\gamma = (hg)^\gamma = h^\gamma g^\gamma,$$

the first and third equalities using properties of  $\mathcal{L}_2$ . □

### Lemma 2.6

- (i) *The kernel of  $\Theta|_{\mathcal{L}_2}$  is isomorphic to the direct product of  $|G/H| - 1$  copies of  $Z(H)$  (the center of  $H$ ).*
- (ii) *(Generators for)  $\text{Ker}(\Theta|_{\mathcal{L}_2})$  can be found in polynomial time.*

PROOF: Let  $\gamma \in \text{Ker}(\Theta|_{\mathcal{L}_2})$ . In particular,  $h^\gamma = h$  for  $h \in H$ . Consider  $Hg \neq H$ . As in the proof of Lemma 2.1, there is some  $k \in H$  such that the action of  $\gamma$  on  $Hg = gH$  is right-multiplication by  $k$ . It suffices then to show  $k \in Z(H)$ . Using Lemma 2.5, for any  $h \in H$ ,

$$g^{-1}hg = (g^{-1}hg)^\gamma = (g^\gamma)^{-1}h^\gamma g^\gamma = (gk)^{-1}h g k = k^{-1}(g^{-1}hg)k.$$

The normality of  $Z(H)$  in  $G$  then implies  $k \in Z(H)$ .

For (ii), we proceed as in Lemma 2.1(ii) but restrict the choice of maps  $x \mapsto xh$  to  $h \in Z(H)$  (or just to a generating set of  $Z(H)$ ).  $\square$

The next two lemmas give necessary conditions on  $(\alpha, \beta) \in \text{Aut}(G/H) \times \text{Aut}(H)$  for it to be liftable to  $\mathcal{L}_2$ .

Recall that  $C$  denotes the centralizer of  $H$  in  $G$ , and  $\iota(x)$  denotes the restriction to  $H$  of the inner automorphism corresponding to  $x$ .

**Lemma 2.7** *Let  $\gamma \in \mathcal{L}_2$ . Suppose that  $\Theta(\gamma) = (\alpha, \beta) \in \text{Aut}(G/H) \times \text{Aut}(H)$ . Then*

(i)  $\alpha$  normalizes  $HC/H$ , and therefore induces an automorphism of  $G/HC$ .

(ii)  $\forall g \in G : \beta^{-1} \iota(HCg) \beta = \iota((HCg)^\alpha)$ .

PROOF: Lemma 2.5 implies that  $C^\gamma = C$ , so that  $(HC)^\gamma = H^\gamma C^\gamma$ , proving (i).

For (ii), first note that  $\forall g \in G, h \in H$ ,

$$h^{\beta^{-1} \iota(g) \beta} = (g^{-1} h^{\beta^{-1}} g)^\beta = (g^\gamma)^{-1} h g^\gamma = h^{\iota(g^\gamma)},$$

the second equality following from Lemma 2.5. In other words,

$$\forall g \in G, \beta^{-1} \iota(g) \beta = \iota(g^\gamma). \quad \square$$

**Lemma 2.8** *Suppose  $(\alpha, \beta) \in \text{Aut}(G/H) \times \text{Aut}(H)$  satisfies (i),(ii) in Lemma 2.7. Then*

$$\forall g \in G : \beta^{-1} (\iota(Hg)) \beta = \iota((Hg)^\alpha). \quad (1)$$

PROOF: For any  $A \subseteq G$ ,  $\iota(AC) = \iota(A)$ . Also, by (i),  $\alpha$  permutes the cosets of  $HC/H$  in  $G/H$  so that  $(HCg)^\alpha = C(Hg)^\alpha$ .  $\square$

**Remark 2.9** This organization may seem convoluted seeing that equation (1) could also be viewed as a direct consequence of Lemma 2.5. Our motive is that, in the process of cutting down to “liftable”  $(\alpha, \beta)$ , our *algorithmic* route runs through (1) after first forcing (i),(ii) of Lemma 2.7.



**Lemma 2.10** Suppose  $(\alpha, \beta) \in \text{Aut}(G/H) \times \text{Aut}(H)$  satisfies property (1). Then  $\Theta^{-1}(\alpha, \beta) \cap \mathcal{L}_2$  is nonempty and can be found in polynomial time.

METHOD: Construct  $\gamma \in \Theta^{-1}(\alpha, \beta)$  as follows.

For  $h \in H$ ,  $h^\gamma := h^\beta$ .

To define  $\gamma$  on a coset  $X$  of  $H$  with  $X \neq H$ . Fix  $a \in X$ . Then, by (1), there is some  $b \in X^\alpha$  such that  $\beta^{-1}\iota(a)\beta = \iota(b)$ . Fix any such  $b$ . For any  $g \in X$ , say  $g = ka$  with  $k \in H$ , set  $g^\gamma := k^\beta b$ . It follows easily that

$$\forall h \in H, (hg)^\gamma = h^\gamma g^\gamma \quad (2)$$

Also

$$\beta^{-1}\iota(g)\beta = \iota(g^\gamma) \quad (3)$$

since  $\beta^{-1}\iota(ka)\beta = \beta^{-1}\iota(k)\beta \beta^{-1}\iota(a)\beta = \iota(k^\beta)\iota(b) = \iota(k^\beta b)$ .

Having established (2) and (3) for  $g$  in each coset  $X$ , they hold for all  $g \in G$ .

Then, for all  $h \in H, g \in G$ ,

$$(gh)^\gamma = (h^{\iota(g)^{-1}}g)^\gamma = h^{\iota(g)^{-1}\beta}g^\gamma = h^{\beta(\beta^{-1}\iota(g)\beta)^{-1}}g^\gamma = h^{\beta\iota(g^\gamma)^{-1}}g^\gamma = g^\gamma h^\gamma.$$

We conclude that  $\gamma \in \mathcal{L}_2$ .

The complete set of preimages of  $(\alpha, \beta)$  is  $\gamma \text{Ker}(\Theta|_{\mathcal{L}_2})$ , for which we apply Lemma 2.6(ii).  $\square$

To summarize the main points of this subsection, we have the following consequence of Lemmas 2.6, 2.7, 2.10.

**Proposition 2.11** Suppose  $\mathcal{M} \leq \text{Aut}(G/H) \times \text{Aut}(H)$  such that property (1) is satisfied by all  $(\alpha, \beta) \in \mathcal{M}$ . Then

- (i)  $\mathcal{M} \leq \Theta(\mathcal{L}_2)$
- (ii)  $\mathcal{L}_2 \cap \Theta^{-1}(\mathcal{M})$  is an extension of  $\mathcal{M}$  by an abelian group.
- (iii) Given  $\mathcal{M}$ ,  $\mathcal{L}_2 \cap \Theta^{-1}(\mathcal{M})$  can be constructed in polynomial time.

PROOF: For (iii),  $\mathcal{L}_2 \cap \Theta^{-1}(\mathcal{M})$  is generated by the lifts of the generators of  $\mathcal{M}$  together with generators of  $\text{Ker}(\Theta|_{\mathcal{L}_2})$ .  $\square$

### 2.3.2 Algorithm

Steps 1-3 cut  $\mathcal{A} \times \mathcal{B}$  to the subgroup  $\mathcal{M}$  consisting of elements that can be lifted to  $\mathcal{L}_2$ .

**Step 1.**  $\mathcal{A} := \mathcal{A}_{HC/H}$ .

METHOD: Viewing  $\mathcal{A} \leq \text{Sym}(G/H)$ , this can be viewed as stabilizing the subset  $HC/H$  of the polynomial-size domain  $G/H$ .  $\square$

By Lemma 2.7(i), Step 1 does not affect  $\Theta(\mathcal{L}_2) \cap (\mathcal{A} \times \mathcal{B})$ .

**Step 2.**  $\mathcal{B} := \mathcal{B}_{\iota(G)}$ .

METHOD: Here we consider  $\mathcal{B} \subseteq \text{Sym}(H)$  acting on  $\text{Aut}(H) \subseteq \text{Sym}(H)$  via conjugation. So, at first glance, this appears to be a normalizer problem for permutation groups. However, the specific reductions of our main problems to AUTLIFTING will reveal that required instances of Step 2 can be handled via set-stabilizers.  $\square$

By Lemma 2.7(ii), Step 2 does not affect  $\Theta(\mathcal{L}_2) \cap (\mathcal{A} \times \mathcal{B})$ .

Step 2 does not yet accomplish the compatibility condition on  $(\alpha, \beta)$  expressed in property (1) and required in Lemma 2.10, but it establishes a structure for getting there.

We now have that  $\mathcal{A} \leq \text{Aut}(G/H)_{HC/H}$ , so there is an induced action

$$\mathfrak{a} : \mathcal{A} \rightarrow \text{Aut}\left(\frac{G}{HC}\right).$$

We also now have  $\mathcal{B}$  acting on  $\iota(G)$  and, since  $\mathcal{B} \leq \text{Aut}(H)$  also normalizes  $\iota(H)$ , there is an induced action

$$\mathfrak{b} : \mathcal{B} \rightarrow \text{Aut}\left(\frac{\iota(G)}{\iota(H)}\right)$$

But there is a natural identification

$$\frac{G}{HC} \simeq \frac{\iota(G)}{\iota(H)}.$$

(For all  $g \in G$ ,  $gHC \leftrightarrow \iota(g)\iota(H)$ .) Via this identification, property (1) is expressible in the form

$$\mathfrak{a}(\alpha) = \mathfrak{b}(\beta).$$

This motivates

**Step 3.**  $\mathcal{M} := \{(\alpha, \beta) \in \mathcal{A} \times \mathcal{B} \mid \mathfrak{a}(\alpha) = \mathfrak{b}(\beta)\}.$

METHOD: By the above, this is a permutation-group intersection problem, solvable for example as a set-stabilizer problem: consider  $\mathfrak{a}(\mathcal{A}) \times \mathfrak{b}(\mathcal{B})$  acting on  $\Omega \times \Omega$  where  $\Omega = G/HC$ ; the object then is to stabilize the diagonal  $\{(\omega, \omega) \mid \omega \in \Omega\}.$   $\square$

By Lemmas 2.7, 2.8 and Proposition 2.11(i),  $\Theta(\mathcal{L}_2) \cap (\mathcal{A} \times \mathcal{B}) = \mathcal{M}.$  Hence,  $\mathcal{L}_2 \cap \Theta^{-1}(\mathcal{A} \times \mathcal{B}) = \mathcal{L}_2 \cap \Theta^{-1}(\mathcal{M}).$  So the next step is

**Step 4.**  $\widehat{\mathcal{L}}_2 := \mathcal{L}_2 \cap \Theta^{-1}(\mathcal{M}).$

METHOD: This is in polynomial time by Proposition 2.11(iii).  $\square$

The final step is

**Step 5.** Find  $\{\gamma \in \widehat{\mathcal{L}}_2 \mid \gamma \in \text{Aut}(G)\}.$

METHOD: As in §2.2.2, this can be expressed as a set-stabilizer problem for the action of  $\widehat{\mathcal{L}}_2$  on  $G \times G \times G.$   $\square$

Thus the computational complexity of our use of AUTLIFTING rests on properties of  $\mathcal{A}$  and  $\mathcal{B}$  that will enable efficient routines for Steps 1, 2, 3, 5. The properties are described in §3.

## 3 Nice groups

### 3.1 A reminder on polynomial-time set stabilizers

In [12], the author proposed an algorithm for finding  $G_\Delta$  where  $G \leq \text{Sym}(\Omega)$  and  $\Delta \subseteq \Omega.$  For the reader's convenience in what follows, we offer a brief description.<sup>1</sup>

To accommodate a recursion, the problem is generalized to cosets.

---

<sup>1</sup>See also [15] for an extended discussion of the divide-and-conquer paradigm for this and other applications.

For a  $G$ -stable subset  $\Pi \subseteq \Omega$  and a coset  $X = Ga$  of  $G$ , define

$$X_{\Delta|\Pi} = \{x \in X \mid (\Delta \cap \Pi)^x = \Delta \cap \Pi^a\}.$$

So  $X_{\Delta|\Pi}$  is either  $\emptyset$  or a right coset of  $G_{\Delta \cap \Pi}$ . If  $\Pi = \Pi_1 \dot{\cup} \Pi_2$  for  $G$ -stable  $\Pi_i$

$$X_{\Delta|\Pi} := (X_{\Delta|\Pi_1})_{\Delta|\Pi_2}$$

If  $G$  acts transitively on  $\Pi$  and  $|\Pi| > 1$ , find a minimal decomposition  $\Pi = \Pi_1 \dot{\cup} \dots \dot{\cup} \Pi_m$  into blocks of imprimitivity and decompose  $G = \bigcup_{1 \leq i \leq |G/H|} Ht_i$  where  $H$  is the kernel of the action of  $G$  on  $\{\Pi_i\}_{1 \leq i \leq m}$ .

$$X_{\Delta|\Pi} := \bigcup_{1 \leq i \leq |G/H|} (Ht_i a)_{\Delta|\Pi}.$$

If  $\Pi = \{\pi\}$  then

$$\text{if } |\Delta \cap \{\pi, \pi^a\}| = 1 \text{ then } X_{\Delta|\Pi} := \emptyset \text{ else } X_{\Delta|\Pi} := X.$$

The computation of  $G_\Delta$  starts with  $\Pi := \Omega$  and  $a := 1$ . The key to the complexity of the method lies in the sizes of the primitive groups arising in the action on the  $m$  blocks. The algorithm will run in polynomial time for an hereditary class of groups if such induced subgroups of  $S_m$  have order  $O(m^{\text{constant}})$ .

Notice that the “set-stabilizer” algorithm actually dealt with cosets. Thus, we can find set-stabilizers for a group  $A$  that has a small-index subgroup  $B$  in a good class: we start by breaking  $A$  into cosets of  $B$ . We will find ourselves in exactly that situation in §3.2.

**Remark 3.1** If we were only given a black-box for set-stabilizers in groups with bounded non-cyclic composition factors, i.e., without knowing the algorithm, we could still use that for the coset problem. It is useful for another purpose in §4.1 to express that as a “set-transporter” problem; namely, given  $\Delta, \Lambda \subseteq \Omega$ , find

$$G_{\Delta \rightarrow \Lambda} = \{g \in G \mid \Delta^g = \Lambda\}.$$

For this, consider the natural action of  $\tilde{G} = G \wr C_2$  on  $\Omega \dot{\cup} \Omega'$ , where  $\Omega'$  is a copy of  $\Omega$ . The desired transporter can be deduced from  $\tilde{G}_{\Delta \dot{\cup} \Lambda'}$  where  $\Lambda'$  is the corresponding copy of  $\Lambda$ . In particular,  $(Ga)_\Delta = G_{\Delta \rightarrow \Delta^{a^{-1}}} a$ .

### 3.2 The groups that turn up

In effect, we will only rely on the polynomial boundedness of primitive solvable groups. However, we deal with groups that are not quite solvable.

**Notation.** For a group  $X$ ,  $\text{Rad}(X)$  is the solvable radical of  $X$ , i.e., the maximum normal solvable subgroup.

**Definition.** Let  $X \leq \text{Sym}(\Omega)$ . We call  $X$  *almost-solvable*<sup>2</sup> if  $|X : \text{Rad}(X)| \leq |\Omega|^2$ .

Almost-solvable groups still enable the polynomial-time divide-and-conquer paradigm. For example,

**Lemma 3.2** *Given an almost-solvable  $G \leq \text{Sym}(\Omega)$  and  $\Delta \subseteq \Omega$ ,  $G_\Delta$  can be found in polynomial time.*

PROOF: By decomposing  $G$  into cosets of  $\text{Rad}(G)$ , we reduce to  $|\Omega|^2$  problems involving solvable groups.  $\square$

**Remark 3.3** Our breakdown to cosets of a nice group is reminiscent of the first use of the set-stabilizer method. When [12] was written, primitive groups in the relevant class, specifically the class of groups with bounded composition factors, were not known to be polynomially bounded. This difficulty was overcome by partitioning the primitive group into cosets of a small-index  $p$ -subgroup. That additional complication soon became unnecessary as polynomial bounds were established first for primitive solvable groups (independently by Pálffy [17] and Wolf [23]), and ultimately for primitive groups in a class that even includes groups with bounded *non-cyclic* composition factors (Babai *et al.* [3]). We point, however, to a subtle difference between what happens in our current passage to cosets of a nice group and the earlier use of that trick. In [12], after passing to cosets of a  $p$ -group acting on the blocks, the divide-and-conquer narrows our window to a single, stabilized block. Since the group acting within the block is not necessarily a  $p$ -group, we may again arrive at a primitive group that requires the cosets-of- $p$ -group decomposition. By contrast, in the present case there is a *single* such decomposition in the lifetime of the set-stabilizer process, i.e., we only visit solvable groups thereafter.

---

<sup>2</sup> The author regrets using a term that has had other meanings. However, the usage of “almost-solvable” is already inconsistent in the literature and he could not think of an *unused* term with similar connotation.

**Remark 3.4** Lemma 3.2 is a weak consequence of the discussion in §3.1. The good subgroup can be of any polynomially-bounded index, need not be normal, and need only be in the broader class described in [3]. However, “almost-solvable” is a good fit for our situation because we next see that the property arises so conveniently. Furthermore, in our present application, one can keep track of small-index normal solvable subgroups as we cut down the group or take preimages, so there is no need even to implement a radical-finder. Nevertheless, we do note that radicals of permutation groups can be found in polynomial-time ([14, Lect. 6], [21, §6.3.1]).

The relevance to our problem is seen in

**Lemma 3.5** *With reference to the groups in §2, suppose  $\mathcal{A} \leq \text{Sym}(G/H)$  and  $\mathcal{B} \leq \text{Sym}(H)$  are almost-solvable.*

- (i) *If  $H$  is solvable, then  $\widehat{\mathcal{L}}_1 \leq \text{Sym}(G)$  is almost-solvable.*
- (ii)  *$\widehat{\mathcal{L}}_2 \leq \text{Sym}(G)$  is almost-solvable.*

PROOF: We have  $|\mathcal{A} \times \mathcal{B} : \text{Rad}(\mathcal{A} \times \mathcal{B})| = |\mathcal{A} : \text{Rad}(\mathcal{A})| \cdot |\mathcal{B} : \text{Rad}(\mathcal{B})| \leq |G/H|^2 |H|^2 = |G|^2$ . Thus, (i) follows from Proposition 2.3(ii).

Using  $\mathcal{M}$  as in §2.3.2 (Steps 3,4),  $\mathcal{M} \leq \mathcal{A} \times \mathcal{B}$  implies  $|\mathcal{M} : \text{Rad}(\mathcal{M})| \leq |\mathcal{A} \times \mathcal{B} : \text{Rad}(\mathcal{A} \times \mathcal{B})| \leq |G|^2$ . Thus (ii) follows from Proposition 2.11(ii).  $\square$

Note, as the methods for AUTLIFTING are to be used repeatedly, the almost-solvability of  $\widehat{\mathcal{L}}_i$  implies that of the subgroup  $\widehat{\mathcal{L}}_i \cap \text{Aut}(G)$ .

For a base case, we need a consequence of the Classification of Finite Simple Groups. Using the fact that any simple group  $T$  can be generated by two elements ([1]), we immediately have  $|\text{Aut}(T)| \leq |T|^2$ . Hence,

**Lemma 3.6** *If  $T$  is simple then  $\text{Aut}(T)$ , viewed as a subgroup of  $\text{Sym}(T)$ , is almost-solvable.*

In fact, it is well known that the Classification yields a stronger bound of the form  $|\text{Aut}(T)| = O(|T| \log |T|)$  (e.g., see [5]). However, the square bound in “almost-solvable” is easier to maintain through our process.

## 4 Automorphisms stabilizing a composition series

Applying the machinery of §§2-3, we offer two polynomial-time Turing reductions of COMP-SERIESAUTO to polynomial-time instances of AUTLIFTING. Moreover, we show that the deepest tool needed in either implementation is set-stabilizer for solvable permutation groups. Aside from reducing to the most basic tool, this extra effort will be useful in a subsequent development of canonical forms. (See also [4] for an indication of how the divide-and-conquer method for set-stabilizer translates to canonical set placement.)

### 4.1 Bottom-up on given series

We go from a solution for  $G_i$  to a solution for  $G_{i-1}$ . The group on top,  $G_{i-1}/G_i$ , is simple and so one can enumerate  $\text{Aut}(G_{i-1}/G_i)$ .

**Remark 4.1** Even for a simple *permutation group*  $T$  given by generators, one can produce generators for  $\text{Aut}(T)$  (which is all that is necessary for some of our subproblems). This follows from Kantor’s demonstration [9] that one can obtain the “natural” representation of  $T$ .

**Remark 4.2** Before proceeding further, we can claim to have already established a polynomial-time solution for this use of AUTLIFTING. That is, by virtue of almost-solvability (§3), there are citable polynomial-time methods for Steps 1,2,3,5 in §2.3. Thus, a message of this subsection is that we do not need the full power of available polynomial-time tools. That leads to speculation that there are more general problems to solve.

Let us consider the steps of the algorithm in §2.3.2. Since there are more interesting things to say about Step 2, we will postpone that discussion and dispense with the other steps.

**Step 1:** Since we always arrive at AUTLIFTING with  $G/H$  simple, either  $HC = G$  or  $HC = H$ . In either case,  $\mathcal{A}$  already normalizes  $HC/H$  so there is nothing more to do.

**Step 3:** This is a group intersection. So it would be in polynomial time if just one group is almost-solvable [12, §4.2] and, as indicated, just a polynomial-time *set-stabilizer* if both groups are almost-solvable, as is the case here. However, in this situation, none of the machinery is even needed because  $\mathcal{A}$  is listable. Note also that this is trivial when  $HC = G$  (which means  $\mathcal{M} = \mathcal{A} \times \mathcal{B}$ ).

**Step 5:** As indicated, this is a set-stabilizer for a group that we now know to be almost-solvable.

Now for **Step 2:**

Three approaches will be indicated, winding up with just set-stabilizers. We state these for a broader class than almost-solvable groups.

For an integer constant  $d > 0$ , let  $\Gamma_d$  denote the class of finite groups all of whose nonabelian composition factors lie in  $S_d$ . Also, bear in mind that the polynomial timing for  $\Gamma_d$  groups immediately extends to situations where we are in possession of a  $\Gamma_d$  subgroup of polynomial index (see §3.) In particular, these methods apply to almost-solvable groups.

PROBLEM (I)

GIVEN:  $X, Y \leq \text{Sym}(\Omega)$  with  $X \in \Gamma_d$ .

FIND:  $X_Y$ . (Where  $X$  is acting on  $\text{Sym}(\Omega)$  via conjugation.)

*Method.* This was shown to be in polynomial-time by Luks and Miyazaki [15].<sup>3</sup> □

In our situation, we are trying to normalize a *polynomial-size*  $Y$  and there is a more elementary approach to this special situation.

PROBLEM (II)

GIVEN:  $X, Y \leq \text{Sym}(\Omega)$  with  $X \in \Gamma_d$  and  $|Y| = O(|\Omega^{\text{const}}|)$ .

FIND:  $X_Y$

METHOD: Let  $\text{Sym}(\Omega)$  act on  $\Omega \times \Omega$  diagonally. Also, for  $s \in \text{Sym}(\Omega)$ , let

$$\Delta_s := \{(\omega, \omega^s) \mid \omega \in \Omega\}.$$

Then for  $y, x \in \text{Sym}(\Omega)$ ,

$$\Delta_y^x = \Delta_{x^{-1}yx}.$$

Thus,  $x$  normalizes  $Y$  iff  $x$  stabilizes the collection  $\{\Delta_y\}_{y \in Y}$ . So, finding  $X_Y$  is a matter of finding the subgroup of  $X$  inducing automorphisms of a hypergraph. Miller [16] has shown that to be in polynomial time for  $X \in \Gamma_d$ . □

Our situation is even more special.

---

<sup>3</sup>While  $\Gamma_d$  is slightly smaller than the class available just for set-stabilization [3], the restriction is still needed for this method.



PROBLEM (III)

GIVEN:  $X, Y \leq \text{Sym}(\Omega)$  with  $X \in \Gamma_d$  and  $|Y| = O(|\Omega|^{\text{const}})$ ;  
 $K < Y$  with  $K \triangleleft \langle X, Y \rangle$  and we are able to list  $\text{Aut}(Y/K)$ .

FIND:  $X_Y$ .

METHOD: Note that  $X_Y = X_{Y/K}$ . For each  $\sigma \in \text{Aut}(Y/K)$ , we find those  $x \in X$  such that conjugation by  $x$  induces  $\sigma$ . This will be case iff

$$\forall y \in Y : x^{-1}yx \in (yK)^\sigma$$

which is the case iff

$$\forall y \in Y, \exists z \in (yK)^\sigma : \Delta_y^x = \Delta_z$$

(See Remark 3.1 for viewing these “set-transporters” as set-stabilizers.) It is feasible to run through all  $y$  and then all  $z$ .  $\square$

To apply the PROBLEM (III) method in our situation,  $Y = \iota(G)$ ,  $K = \iota(H)$ . Furthermore, we are only concerned with the case  $HC = H$ , else  $Y = \iota(G) = \iota(H)$  which is already normalized by  $\mathcal{B}$ . Thus  $Y/K$  is simple. So we can not only list  $\text{Aut}(Y/K)$  but we can even shorten the process by checking that  $x$  conjugates correctly on just  $y_1, y_2 \in Y$ , where  $y_1K, y_2K$  generate  $Y/K$ .

A further savings on the number of set-stabilizer calls can be realized by using a quotient-group method of Kantor and Luks [10, problem P7(ii)]. Finding the  $x \in X$  such that  $(yK)^x = (yK)^\sigma$  can be accomplished with a single set-stabilizer.

## 4.2 Top-down on a refinement of a characteristic series

Recall that a subgroup  $H$  of a group  $G$  is called *characteristic* if it is invariant under  $\text{Aut}(G)$ . A group with no proper characteristic subgroups is called *characteristically simple* and is necessarily a product of isomorphic simple groups. A *characteristic series* in  $G$  is a chain  $G = K_0 \triangleright K_1 \triangleright K_2 \triangleright \cdots \triangleright K_r = \mathbf{1}$  for which each  $K_i$  is characteristic in  $G$ .

A characteristic series can be constructed with characteristically simple quotients  $K_i/K_{i+1}$  even if  $G$  is a permutation group given by generators; see, e.g., [10]. For groups given by a Cayley table, the construction is elementary: find the minimal normal subgroups of  $G$  by considering the normal closures of all elements; these will each be characteristically simple, so select those whose simple factors are of designated type and let them generate the subgroup  $K$ . Continue the process with  $G/K$ , etc.

**Lemma 4.3** *Without loss of generality, we may assume that the series  $G_0, G_1, G_2, \dots, G_m$  in COMPSERIESAUTO has a characteristic subseries*

$$G = K_0 \triangleright K_1 \triangleright K_2 \triangleright \cdots \triangleright K_r = \mathbf{1}$$

where each  $K_i/K_{i+1}$  is characteristically simple.<sup>4</sup>

PROOF: We are given a composition series

$$G = G_0 \triangleright G_1 \triangleright G_2 \triangleright \cdots \triangleright G_m = \mathbf{1}.$$

As indicated above, we construct a *characteristic* series  $G = K_0, K_1, K_2, \dots, K_r = \mathbf{1}$  with characteristically simple quotients  $K_i/K_{i+1}$ . Refine the series between each  $K_i$  and  $K_{i+1}$  by inserting

$$K_i = (K_i \cap G_0)K_{i+1} \supseteq (K_i \cap G_1)K_{i+1} \supseteq (K_i \cap G_2)K_{i+1} \supseteq \cdots \supseteq (K_i \cap G_m)K_{i+1} = K_{i+1}$$

and eliminate duplicates. We again have a composition series and automorphisms stabilizing the original series will stabilize this new one. Having computed the automorphisms stabilizing the new series, we have an almost-solvable group, so cutting down the result to stabilize the original series is done with set stabilizers.  $\square$

*Method for Theorem 1.2.*

Having reset the series as in Lemma 4.3, successively compute the appropriate subgroup of  $\text{Aut}(K_0/K_i)$ , where  $(K_i)_i$  is the embedded normal series as in Lemma 4.3.

In the base case  $K_0/K_0$  is trivial.

For the inductive step, the call to `AUTLIFTING` involves  $K_0/K_{i+1}$  and its normal subgroup  $K_i/K_{i+1}$ . We arrive with the inductive input  $\mathcal{A} \leq \text{Aut}(K_0/K_i)$ . The group  $\mathcal{B}$  should consist of the automorphisms of the semisimple group  $H := K_i/K_{i+1}$  that fix the composition series induced on this section. If  $H$  is nonabelian,  $\mathcal{B}$  is the direct product of the automorphism groups of the simple factors. If  $H$  is a product of cyclic groups of prime order  $p$ ,  $\mathcal{B}$  can be viewed as the upper triangular matrices over  $\text{GF}(p)$ .

Using the  $\mathcal{L}_2$  method at each stage, the procedure is in polynomial time for *all groups*, but that seemed to require hypergraph stabilizer in Step 2. So let us revisit that step.

Suppose  $H$  is nonabelian. Then  $\mathcal{B}$  (which now fixes the simple factors) is of polynomial size, e.g.  $|\mathcal{B}| \leq |H|^2$  in which case Step 2 can be carried out by testing each element of  $\mathcal{B}$ . This is not the case if  $H$  is abelian, but then we simply revert to the  $\mathcal{L}_1$  method of §2.2 for this round.  $\square$

## 5 Isomorphism matching fixed composition series

We prove Theorem 1.1 via a familiar technique in isomorphism studies, applying the automorphism-group result to finding isomorphisms. For example, an algorithm for finding automorphism groups of graphs will find the isomorphisms between two connected graphs

---

<sup>4</sup> The Wagner-Rosenbaum composition series are already of the special type.

$X_1, X_2$  by finding the automorphism group of the disjoint union  $X_1 \dot{\cup} X_2$ . An analogous construction works here.

We are given composition series

$$G_1 = G_{1,0} \triangleright G_{1,1} \triangleright G_{1,2} \triangleright \cdots \triangleright G_{1,m} = \mathbf{1}$$

$$G_2 = G_{2,0} \triangleright G_{2,1} \triangleright G_{2,2} \triangleright \cdots \triangleright G_{2,m} = \mathbf{1}$$

Form the single subnormal series

$$G_1 \times G_2 = G_{1,0} \times G_{2,0} \triangleright G_{1,1} \times G_{2,1} \triangleright G_{1,2} \times G_{2,2} \triangleright \cdots \triangleright G_{1,m} \times G_{2,m} = \mathbf{1} \times \mathbf{1}.$$

We directly accommodate the setup of §4.1 to this situation. (The method of §4.2 could be adapted as well.) We now want the automorphisms of  $G_1 \times G_2$  that not only fix the terms in this series but, in doing so, fix or switch the factors. If, for any  $i$ , we find that there are no relevant automorphisms of  $G_{1,i} \times G_{2,i}$  that switch factors, we exit with a negative response to COMP SERIES ISO.

Calls to AUT LIFTING will now have  $G/H$  as the product of two isomorphic simple groups. For  $\mathcal{A}$  we take the automorphisms that fix or switch the factors. (That would always be the case if the simple groups are nonabelian.)

The rest of the discussion, including the various methods for Step 2, proceed as before.

**Remark 5.1** A trivial technicality. Our weak version of Lemma 3.6 does not quite say  $\text{Aut}(T \times T) \leq \text{Sym}(T \times T)$  is almost-solvable for simple nonabelian  $T$  since  $|\text{Aut}(T \times T)| = 2|\text{Aut}(T)|^2$ . As already noted, Lemma 3.6 could be strengthened, but it is clear that the theory is unaffected by an extra factor of 2.

#### ACKNOWLEDGEMENTS

The author is delighted to acknowledge communications with Laci Babai and James Wilson that stimulated the development of these ideas and improved their exposition.

## References

- [1] M. Aschbacher and R. Guralnick, *Some applications of the first cohomology group*, J. Algebra **90** (1984), 446–460.
- [2] L. Babai, Personal communication (2012).
- [3] L. Babai, P. J. Cameron and P. P. Pálffy, *On the orders of primitive groups with restricted nonabelian composition factors*, J. Algebra **79** (1982), 161–168.

- [4] L. Babai and E. M. Luks, *Canonical labeling of graphs*, Proc. 15th ACM Symp. on Theory of Computing (STOC), 1983, 171–183.
- [5] J. H. Conway, R. T. Curtis, S. P. Norton, R. A. Parker, and R. A. Wilson, *Atlas of Finite Groups: Maximal Subgroups and Ordinary Characters for Simple Groups*, Oxford University Press 1985
- [6] J. D. Dixon and B. Mortimer, *Permutation Groups*, Springer 1996.
- [7] V. Felsch and J. Neubüser, *On a programme for the determination of the automorphism group of a finite group*, Computational Problems in Abstract Algebra, (J. Leech, ed.) Proc. Conf. Comp. Alg, Oxford 1967, Pergamon 1970, 59–60.
- [8] D. F. Holt, B. Eick and E. A. O’Brien, *Handbook of Computational Group Theory, Discrete Mathematics and its Applications*, Chapman & Hall, 2005,
- [9] W. M. Kantor, *Sylow’s theorem in polynomial time*, J. Comput. System Sci. **30** (1985), 359–394.
- [10] W. M. Kantor and E. M. Luks, *Computing in quotient groups*, Proc. 22nd ACM Symp. on Theory of Computing (STOC), 1990, 524–534.
- [11] R. J. Lipton, L. Snyder, and Y. Zalcstein, *The Complexity of Word and Isomorphism Problems for Finite Groups*, Yale CS Technical Report TR091, Defense Technical Information Center 1977.
- [12] E. M. Luks, *Isomorphism of graphs of bounded valence can be tested in polynomial time*, J. Comput. System Sci. **25** (1982), 42–65.
- [13] \_\_\_\_\_ *Lectures on polynomial-time computation in groups*, Technical Report NU-CCS-90-16, Northeastern University, 1990. (<http://www.cs.uoregon.edu/~luks/northeasterncourse.pdf>)
- [14] \_\_\_\_\_, *Permutation groups and polynomial-time computation*, Groups and Computation, Piscataway, N.J., Oct. 7–10, 1991 (L. Finkelstein and W. M. Kantor, eds.), DIMACS Ser. Discrete Math. Theoret. Comput. Sci., vol. 11, Amer. Math. Soc., Providence, R.I., 1993, 139–175.
- [15] E. M. Luks and T. Miyazaki, *Polynomial-time normalizers*, Discrete Math. and Theoret. Comput. Sci., **13** (2011), 61–96.
- [16] G. L. Miller, *Isomorphism of graphs which are pairwise  $k$ -separable*, Inform. and Control **56** (1983), 21–33.
- [17] P. P. Pálffy, *A polynomial bound for the orders of primitive solvable groups*, J. Algebra, **77** (1982), 127–137.

- [18] D. Rosenbaum, *Breaking the  $n^{\log n}$  barrier for solvable-group isomorphism*, Proc. 24th ACM-SIAM Symp. Disc. Algorithms, (2013), 1054–1073.
- [19] \_\_\_\_\_, *Bidirectional collision detection and faster algorithms for isomorphism problems*, arXiv:1304.3935 [cs.DS]
- [20] D. Rosenbaum and F. Wagner, *Beating the Generator-Enumeration Bound for  $p$ -Group Isomorphism*, Theoret. Comput. Sci., **593** (2015) 16–25.
- [21] Á. Seress, *Permutation group algorithms*, Cambridge Tracts in Mathematics, 152. Cambridge University Press, 2003.
- [22] F. Wagner, *On the complexity of group isomorphism*, Electronic Colloq. on Computational Complexity, **18**, 2011.
- [23] T. R. Wolf, *Solvable and nilpotent subgroups of  $GL(n, q^m)$* , Can. J. Math. **34** (1982), 1097–1111.